NoSQL

vs. SQL

micro-23

Aliaksei Bialiauski

Designed in LaTeX

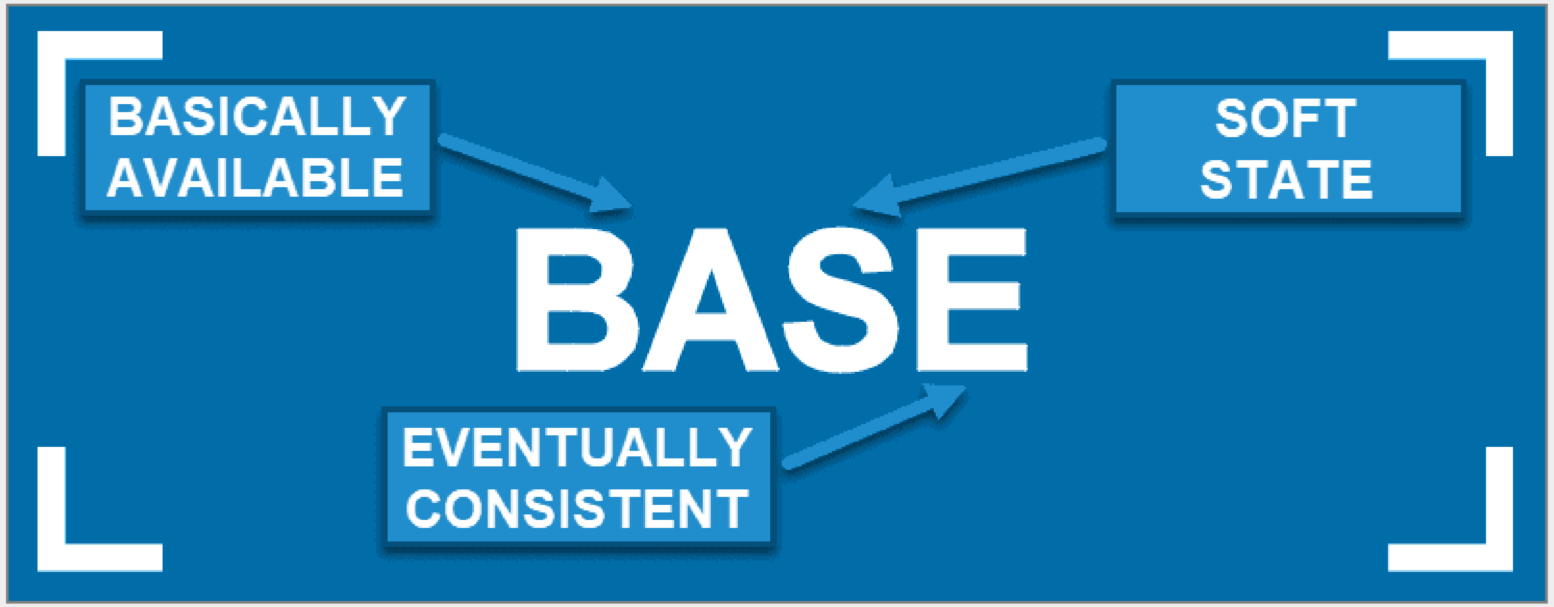
NoSQL Types

Indexes

Database networking

How to Choose the Right Database?

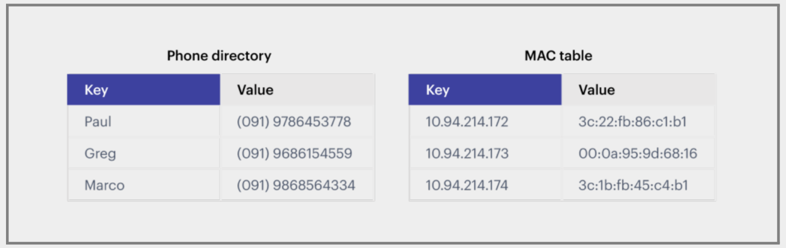
BASE


BASICALLY AVAILABLE
SOFT STATE
BASE
EVENTUALLY CONSISTENT

Chapter #1:

NoSQL Types

Key-Value

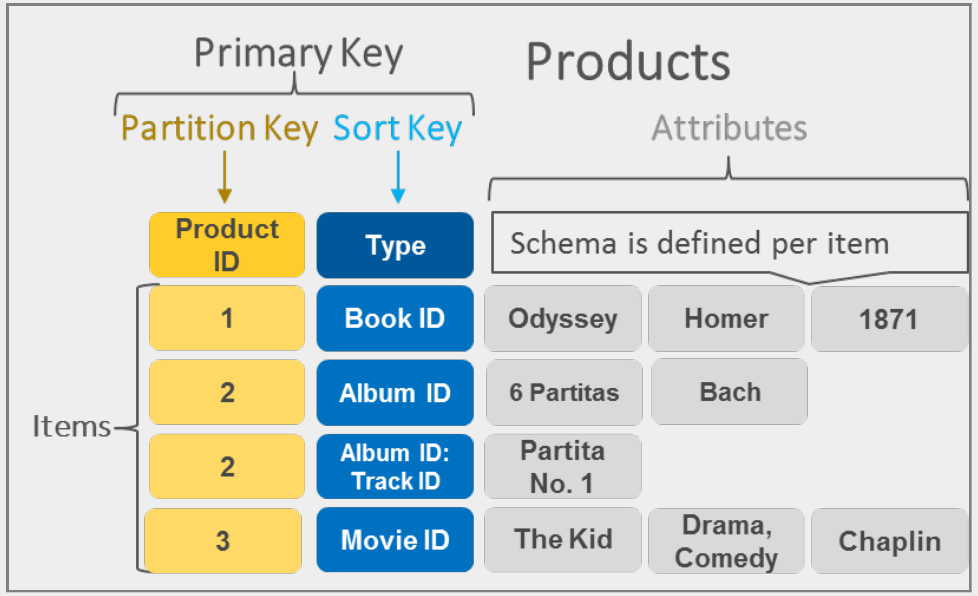


Phone directory

Key	Value
Paul	(091) 9786453778
Greg	(091) 9686154559
Marco	(091) 9868564334

MAC table

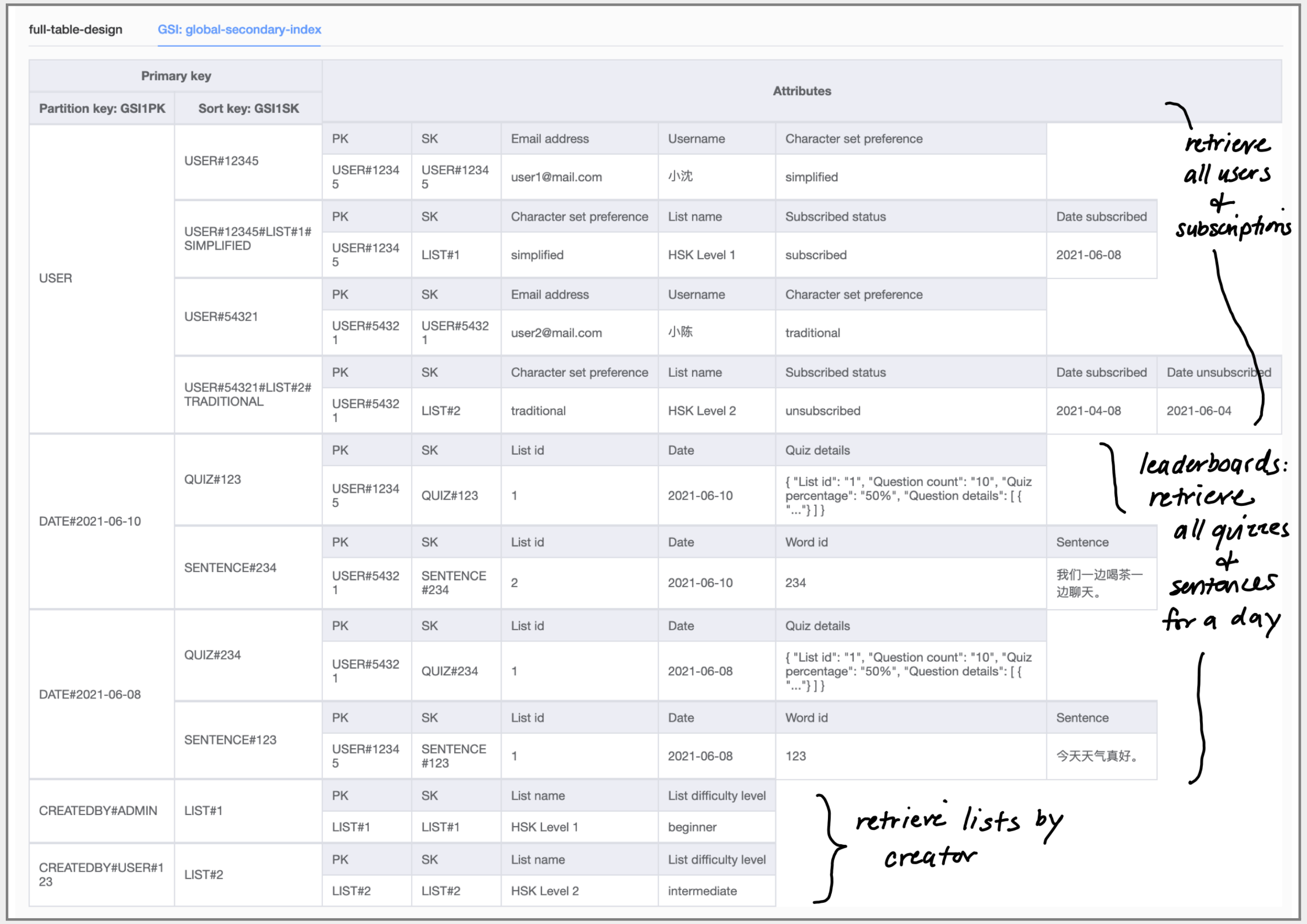
Key	Value
10.94.214.172	3c:22:fb:86:c1:b1
10.94.214.173	00:0a:95:9d:68:16
10.94.214.174	3c:1b:fb:45:c4:b1



Product ID	Type			
1	Book ID	Odyssey	Homer	1871
2	Album ID	6 Partitas	Bach	
2	Album ID: Track ID	Partita No. 1		
3	Movie ID	The Kid	Drama, Comedy	Chaplin

E.g. Redis, AWS DynamoDB

Single Table Design



full-table-design   GSI: global-secondary-index

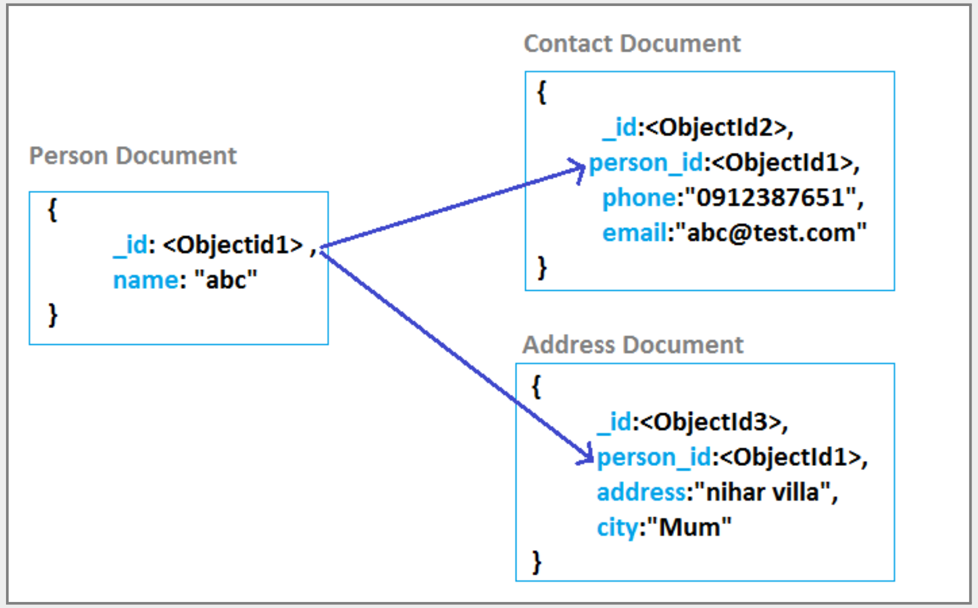
Primary key		Attributes						
Partition key: GSI1PK	Sort key: GSI1SK							
USER	USER#12345	PK	SK	Email address	Username	Character set preference		
		USER#12345	USER#12345	user1@mail.com	小沈	simplified		
	USER#12345#LIST#1#SIMPLIFIED	PK	SK	Character set preference	List name	Subscribed status	Date subscribed	
		USER#12345	LIST#1	simplified	HSK Level 1	subscribed	2021-06-08	
	USER#54321	PK	SK	Email address	Username	Character set preference		
		USER#54321	USER#54321	user2@mail.com	小陈	traditional		
	USER#54321#LIST#2#TRADITIONAL	PK	SK	Character set preference	List name	Subscribed status	Date subscribed	Date unsubscribed
		USER#54321	LIST#2	traditional	HSK Level 2	unsubscribed	2021-04-08	2021-06-04
DATE#2021-06-10	QUIZ#123	PK	SK	List id	Date	Quiz details		
		USER#12345	QUIZ#123	1	2021-06-10	{ "List id": "1", "Question count": "10", "Quiz percentage": "50%", "Question details": [{ "..."}] }		
	SENTENCE#234	PK	SK	List id	Date	Word id	Sentence	
		USER#54321	SENTENCE#234	2	2021-06-10	234	我们一边喝茶一边聊天。	
DATE#2021-06-08	QUIZ#234	PK	SK	List id	Date	Quiz details		
		USER#54321	QUIZ#234	1	2021-06-08	{ "List id": "1", "Question count": "10", "Quiz percentage": "50%", "Question details": [{ "..."}] }		
	SENTENCE#123	PK	SK	List id	Date	Word id	Sentence	
		USER#12345	SENTENCE#123	1	2021-06-08	123	今天天气真好。	
CREATEDBY#ADMIN	LIST#1	PK	SK	List name	List difficulty level			
		LIST#1	LIST#1	HSK Level 1	beginner			
CREATEDBY#USER#123	LIST#2	PK	SK	List name	List difficulty level			
		LIST#2	LIST#2	HSK Level 2	intermediate			

retrieve all users & subscriptions

leaderboards: retrieve all quizzes & sentences for a day

retrieve lists by creator

Document databases



Person Document

```
{
        _id: <Objectid1> ,
        name: "abc"
}
```

Contact Document

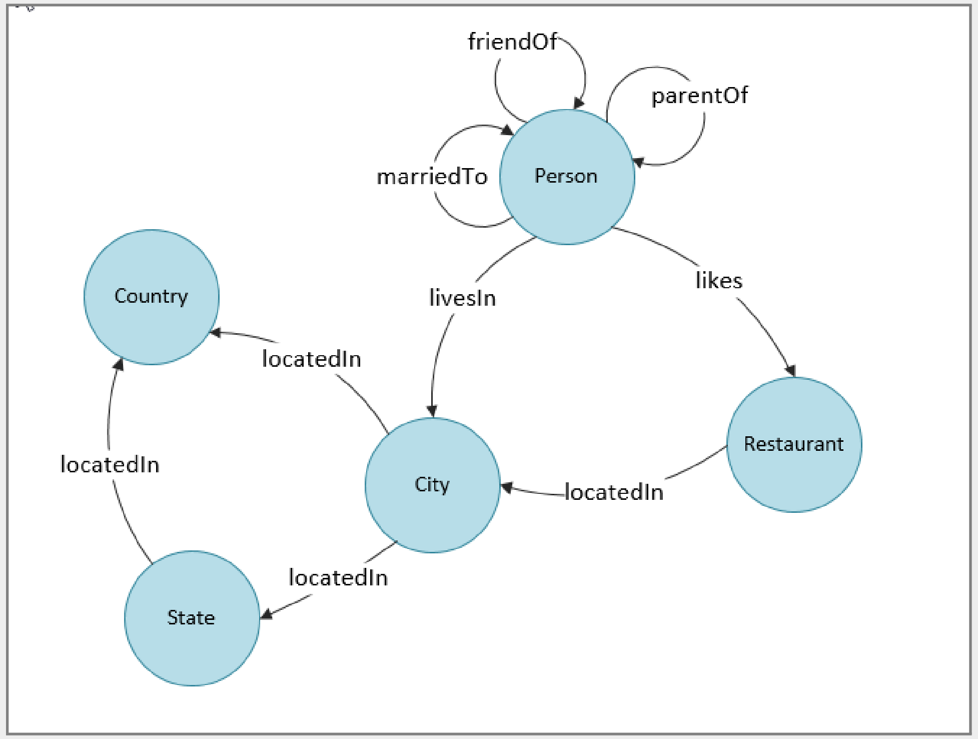
```
{
       _id:<ObjectId2>,
     person_id:<ObjectId1>,
       phone:"0912387651",
       email:"abc@test.com"
}
```

Address Document

```
{
       _id:<ObjectId3>,
       person_id:<ObjectId1>,
       address:"nihar villa",
       city:"Mum"
}
```

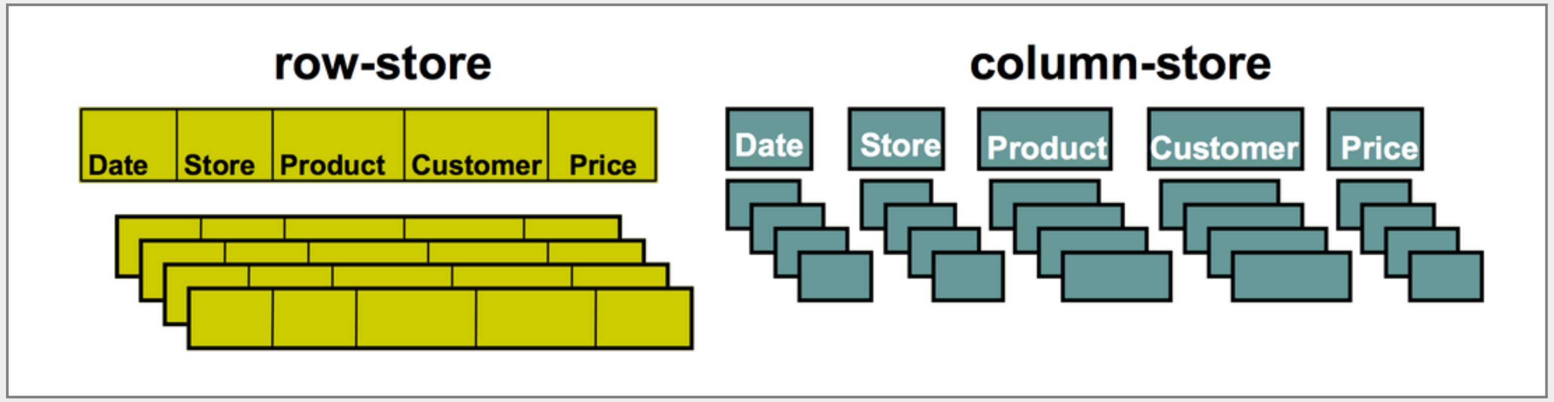
E.g. MongoDB

Graph



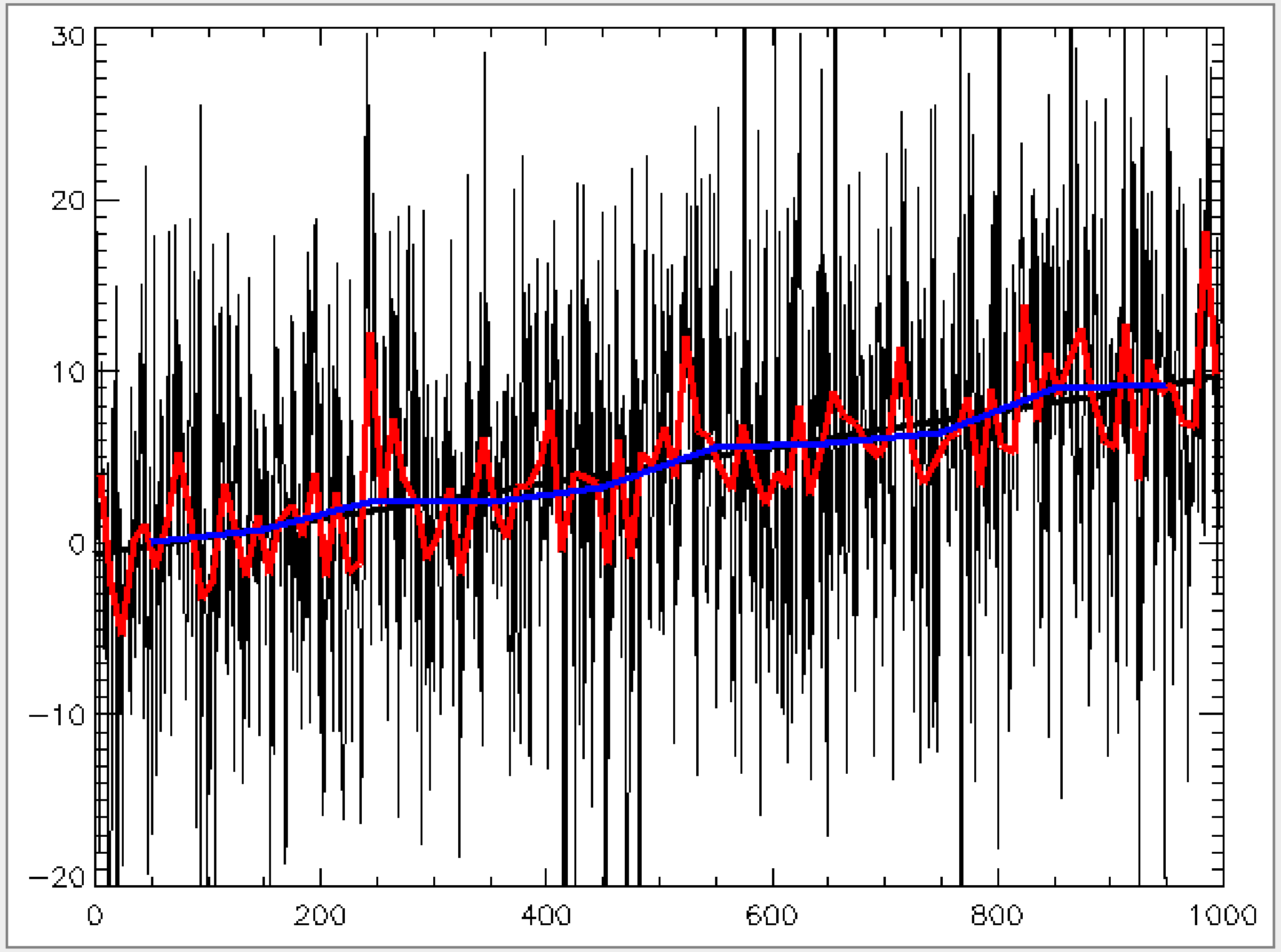
E.g. Neo4j

Wide columnar



E.g. Apache Cassandra

For time-series data and *OLTP*



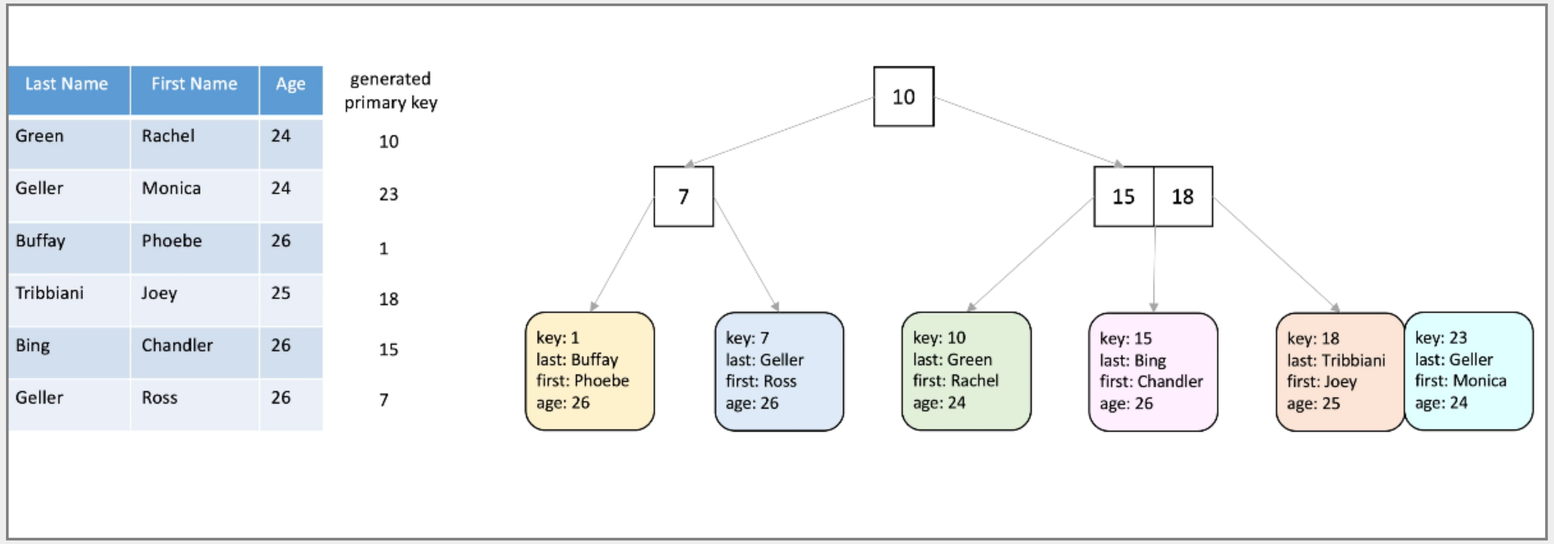
Chapter #2:

Indexes

B-tree index

Self-balancing tree data structure

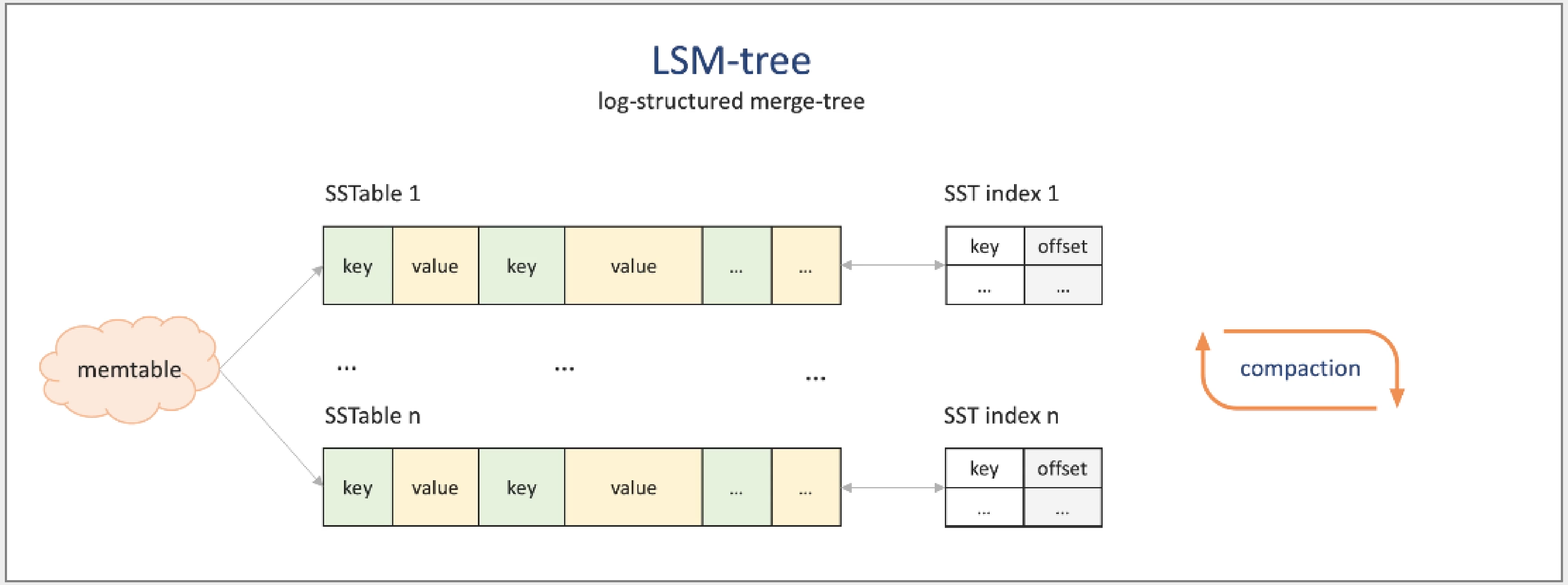
insert, delete, search - *O(log(n))*



Last Name	First Name	Age	generated primary key
Green	Rachel	24	10
Geller	Monica	24	23
Buffay	Phoebe	26	1
Tribbiani	Joey	25	18
Bing	Chandler	26	15
Geller	Ross	26	7

10
7
15
18
key: 1 last: Buffay first: Phoebe age: 26
key: 7 last: Geller first: Ross age: 26
key: 10 last: Green first: Rachel age: 24
key: 15 last: Bing first: Chandler age: 26
key: 18 last: Tribbiani first: Joey age: 25
key: 23 last: Geller first: Monica age: 24

Log Structured Merge-tree



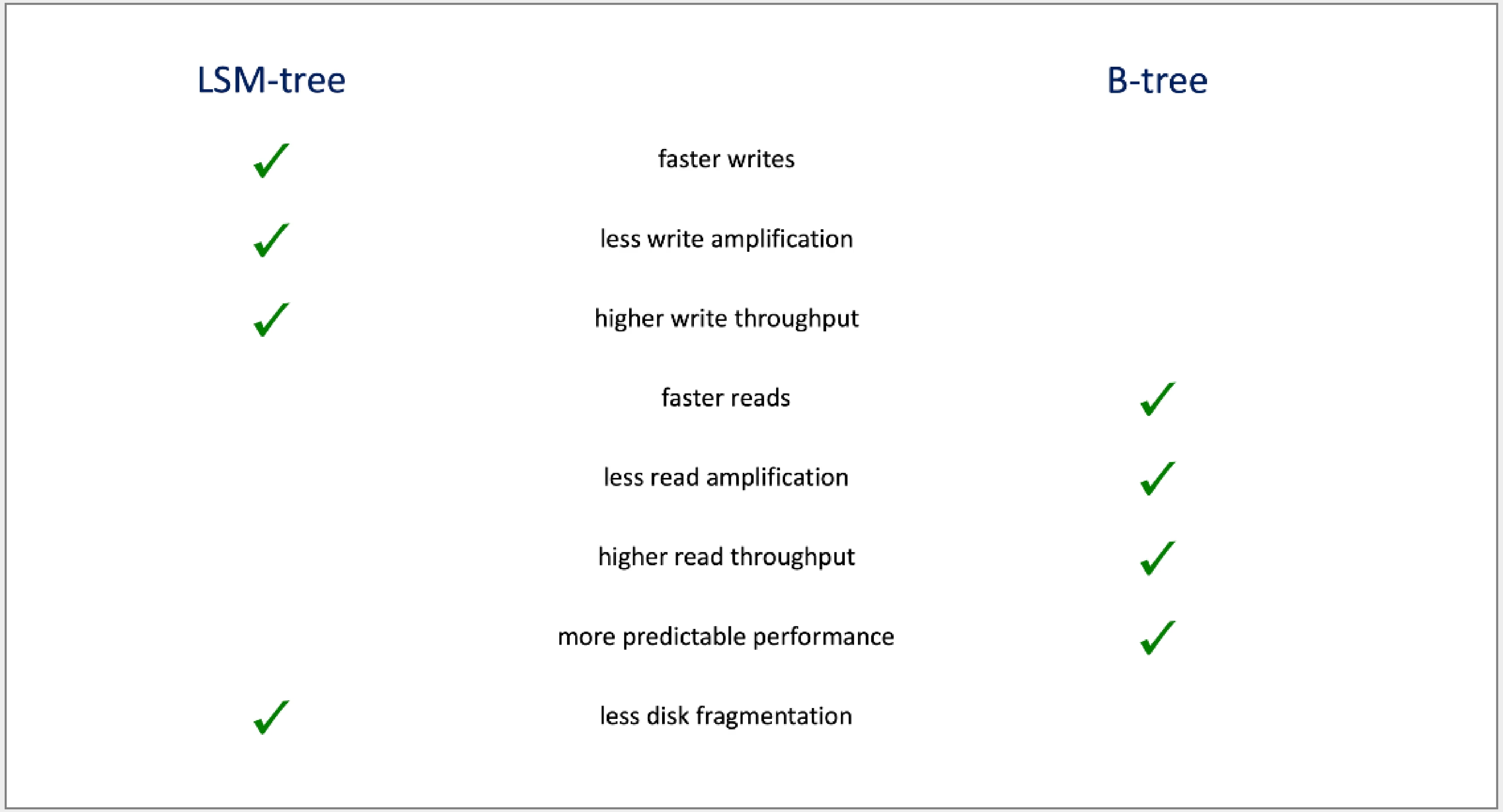
LSM vs. B-tree

LSM

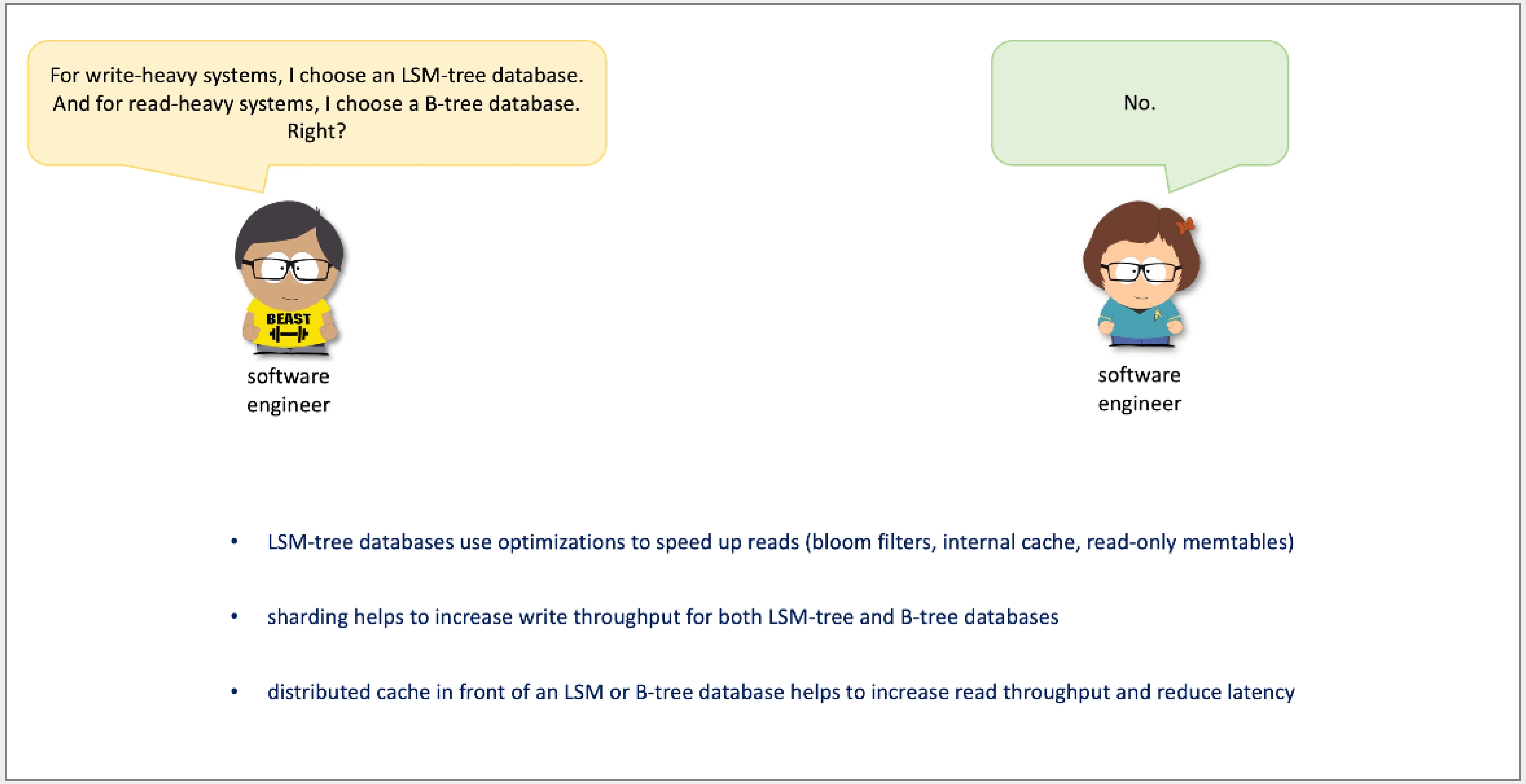
- Google Bigtable
- Apache HBase
- Apache Cassandra
- Influx DB

B-tree

- MySQL
- PostgreSQL
- Apache CouchDB
- DynamoDB



LSM-tree		B-tree
✓	faster writes	
✓	less write amplification	
✓	higher write throughput	
	faster reads	✓
	less read amplification	✓
	higher read throughput	✓
	more predictable performance	✓
✓	less disk fragmentation	



For write-heavy systems, I choose an LSM-tree database.
And for read-heavy systems, I choose a B-tree database.
Right?

No.

software engineer

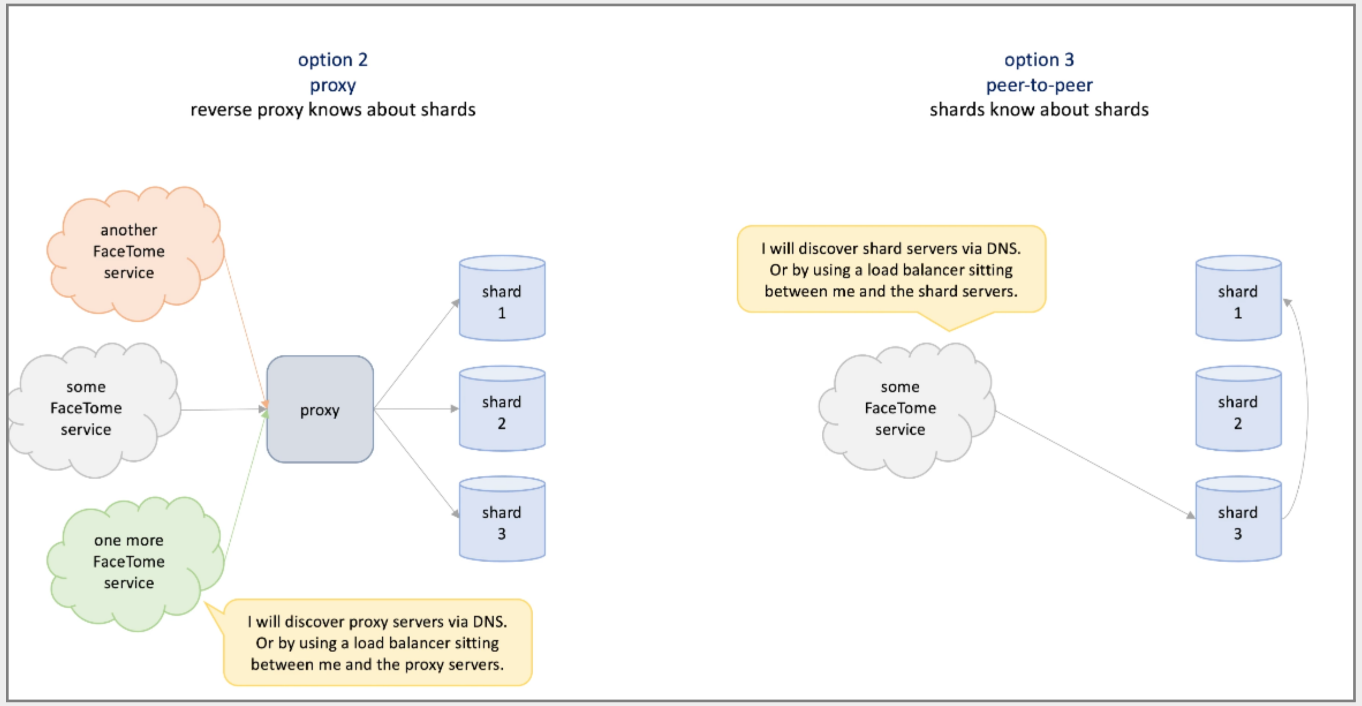
software engineer

- LSM-tree databases use optimizations to speed up reads (bloom filters, internal cache, read-only memtables)
- sharding helps to increase write throughput for both LSM-tree and B-tree databases
- distributed cache in front of an LSM or B-tree database helps to increase read throughput and reduce latency

Chapter #3:

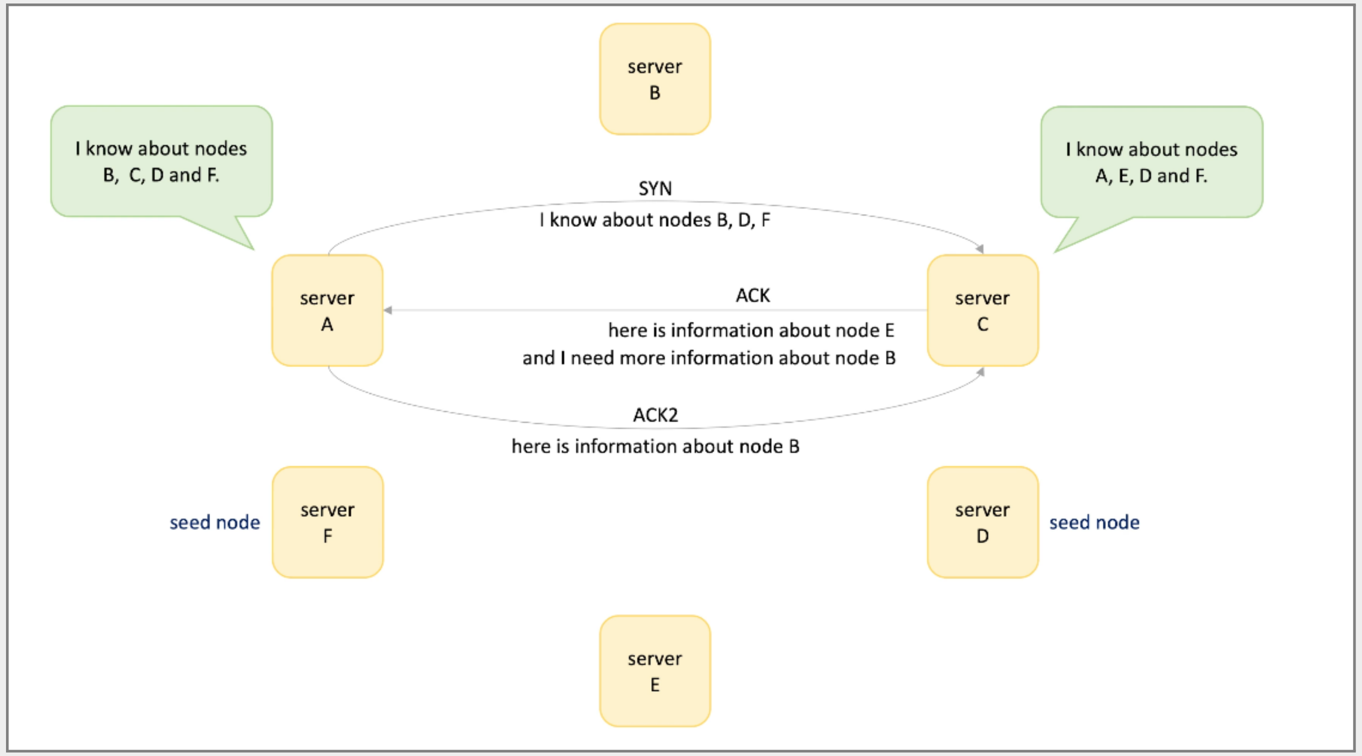
Database networking

Request Routing


option 2
proxy
reverse proxy knows about shards
another FaceTome service
some FaceTome service
one more FaceTome service
proxy
shard 1
shard 2
shard 3
I will discover proxy servers via DNS. Or by using a load balancer sitting between me and the proxy servers.
option 3
peer-to-peer
shards know about shards
I will discover shard servers via DNS. Or by using a load balancer sitting between me and the shard servers.
some FaceTome service
shard 1
shard 2
shard 3

E.g. MySQL, PostgreSQL, MongoDB

Gossip protocol



E.g. Riak, Cassandra, Dynamo

Chapter #4:

How to Choose the Right Database?

Durability: Can We Loose Data?

FAQ: Amazon S3 is designed to provide 99.999999999% (11 9's) of data durability of objects over a given year. This durability level corresponds to an average annual expected loss of 0.000000001% of objects. For example, if you store 10,000,000 objects with Amazon S3, you can on average expect to incur a loss of a single object once every 10,000 years."

ACID

Atomicity: everything or nothing

Consistency: invariants are in place

Isolation: concurrent or sequential

Durability: completed transactions → non-volatile memory

Performance

Queries Profiling & Optimization

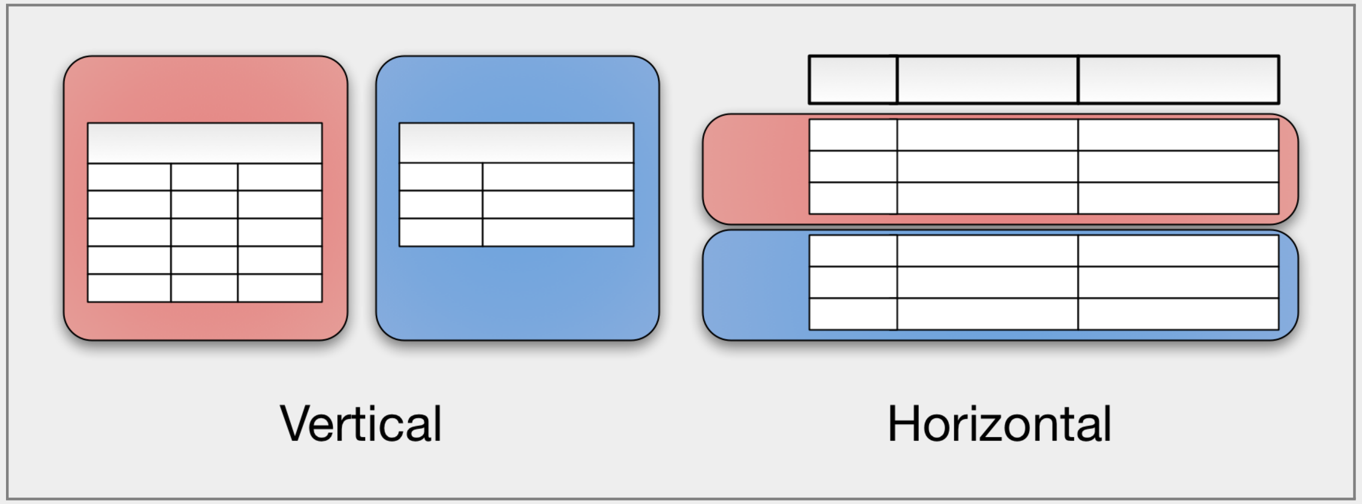
Denormalization

Caching

Scalability

Vertical vs. Horizontal Scalability

Sharding vs. Master-Slave Replication



Application Layer Support

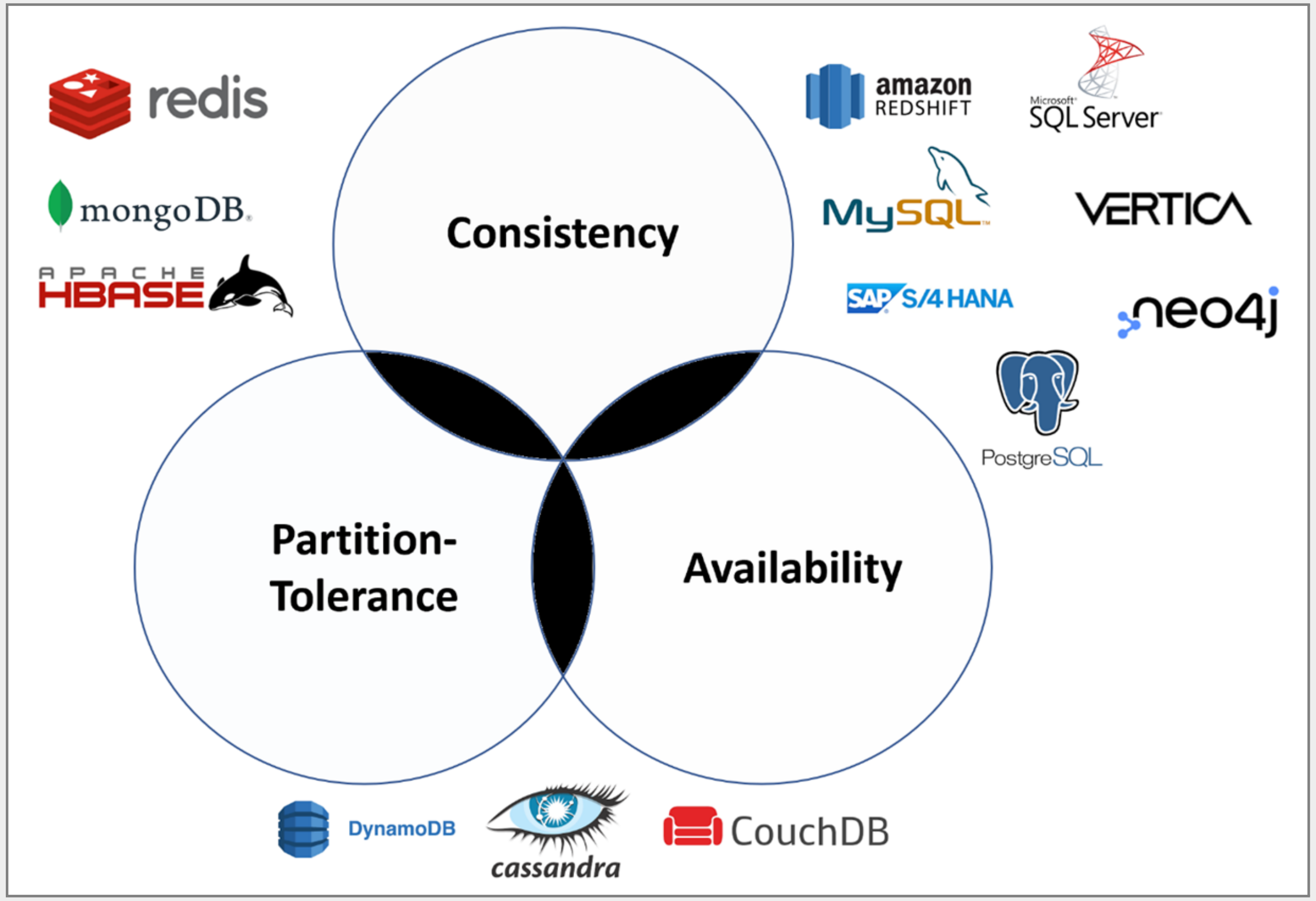
Is it open source?

How mature is the library?

Is it a thin driver or ORM-ish framework?

Is the API open?

CAP


Consistency
Partition-Tolerance
Availability
redis
mongoDB
APACHE HBASE
amazon REDSHIFT
Microsoft SQL Server
MySQL
VERTICA
SAP S/4 HANA
neo4j
PostgreSQL
DynamoDB
cassandra
CouchDB